

[illegible]

3

Sy

MT

MT
MT

MT

MT

MT
MTMT
MTMT
MTMT
MT

MT
MT

MT
MT

MT

MT

MT
MT

MI
MT

MT
MT

MT
MT

MT
MT

MT
MT

MT

111
227

MT

MT
MT

MT

MT

M1
M1MT
MT

M1

M1

10

1

10

100

10

1

OT:
2-

```

000000    TTTTTTTT    SSSSSSSS    PPPPPPPP    000000    WW        WW        GGGGGGGG    LL        UU        UU
000000    TTTTTTTT    SSSSSSSS    PPPPPPPP    000000    WW        WW        GGGGGGGG    LL        UU        UU
00          00          TT          SS          PP          PP          00          00          WW        WW        GG        LL        UU        UU
00          00          TT          SS          PP          PP          00          00          WW        WW        GG        LL        UU        UU
00          00          TT          SS          PP          PP          00          00          WW        WW        GG        LL        UU        UU
00          00          TT          SS          PP          PP          00          00          WW        WW        GG        LL        UU        UU
00          00          TT          SS          PP          PP          00          00          WW        WW        GG        LL        UU        UU
00          00          TT          SS          PP          PP          00          00          WW        WW        GG        LL        UU        UU
00          00          TT          SS          PP          PP          00          00          WW        WW        GG        LL        UU        UU
00          00          TT          SS          PP          PP          00          00          WWWW      WWWW      GG        GG        LL        UU        UU
00          00          TT          SS          PP          PP          00          00          WWWW      WWWW      GG        GG        LL        UU        UU
000000    TT          SSSSSSSS    PP          000000    WW        WW        GGGGGG    LLLLLLLLLL    UUUUUUUUUU
000000    TT          SSSSSSSS    PP          000000    WW        WW        GGGGGG    LLLLLLLLLL    UUUUUUUUUU
...
LL          IIIIIIII    SSSSSSSS
LL          IIIIIIII    SSSSSSSS
LL          II          SS
LL          II          SS
LL          II          SS
LL          II          SS
LL          II          SSSSSS
LL          II          SSSSSS
LL          II          SS
LL          II          SS
LL          II          SS
LL          II          SS
LLLLLLLLLLL IIIIIIII SSSSSSSS
LLLLLLLLLLL IIIIIIII SSSSSSSS

```


(2) 46
(3) 82

DECLARATIONS
OTSS\$POWGLU - G_floating ** unsigned integer power routine

OTSS\$POWGLU
1-001

- G_floating ** unsigned integer power r 16-SEP-1984 02:00:02 VAX/VMS Macro V04-00 Page 1
6-SEP-1984 11:28:18 [MTHRTL.SRC]OTSPOWGLU.MAR;1 (1)

```
0000 1 .TITLE OTSS$POWGLU - G_floating ** unsigned integer power routine
0000 2 .IDENT /1-001/ ; File: OTSPOWGLU.MAR Edit: SBL1001
0000 3
0000 4 :
0000 5 :*****
0000 6 :*
0000 7 :* COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 8 :* DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 9 :* ALL RIGHTS RESERVED.
0000 10 :*
0000 11 :* THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 12 :* ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 13 :* INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 14 :* COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 15 :* OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 16 :* TRANSFERRED.
0000 17 :*
0000 18 :* THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 19 :* AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 20 :* CORPORATION.
0000 21 :*
0000 22 :* DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 23 :* SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 24 :*
0000 25 :*
0000 26 :*****
0000 27 :
0000 28 :
0000 29 :++
0000 30 : FACILITY: Language support procedures, Mathematics division
0000 31 :
0000 32 : ABSTRACT:
0000 33 :
0000 34 : This module contains OTSS$POWDLU, a procedure which raises a
0000 35 : G_floating value to an unsigned integer power.
0000 36 :
0000 37 : ENVIRONMENT: Runs at any access mode, AST Reentrant
0000 38 :
0000 39 : AUTHOR: Steven B. Lionel, CREATION DATE: 22-JUL-1981
0000 40 :
0000 41 : MODIFIED BY:
0000 42 :
0000 43 : 1-001 - Original. SBL 22-JUL-1981
0000 44 :--
```



```

0000 46 .SBTTL DECLARATIONS
0000 47 :
0000 48 : LIBRARY MACRO CALLS:
0000 49 :
0000 50 $SSDEF ; System error codes
0000 51 $CHFDEF ; Define condition handler symbols.
0000 52 $SFDEF ; Define stack frame symbols.
0000 53 $PSLDEF ; Define program status longword
0000 54 ; symbols.
0000 55 :
0000 56 : EXTERNAL DECLARATIONS:
0000 57 :
0000 58 .DSABL GBL ; Force all external symbols to be declared
0000 59 .EXTRN MTH$K_UNDEXP ; Undefined exponentiation
0000 60 .EXTRN MTH$K_FLOOVEMAT ; Floating overflow in Math Library
0000 61 .EXTRN MTH$K_FLOUNDMAT ; Floating underflow in Math Library
0000 62 .EXTRN MTH$$SIGNAL ; Math error routine
0000 63 :
0000 64 : MACROS:
0000 65 :
0000 66 : NONE
0000 67 :
0000 68 : EQUATED SYMBOLS:
0000 69 :
0000 70 : NONE
0000 71 :
0000 72 : OWN STORAGE:
0000 73 :
0000 74 : NONE
0000 75 :
0000 76 : PSECT DECLARATIONS:
0000 77 :
00000000 78 .PSECT _OTSS$CODE PIC, USR, CON, REL, LCL, SHR, -
0000 79 EXE, RD, NOWRT, LONG
0000 80

```

```

0000 82      .SBTTL OTSS$POWGLU - G_floating ** unsigned integer power routine
0000 83      :++
0000 84      : FUNCTIONAL DESCRIPTION:
0000 85      :
0000 86      :   Raise a G_floating value to an unsigned integer power giving
0000 87      :   a G_floating result.
0000 88      :
0000 89      :   The result is given by:
0000 90      :
0000 91      :   base      exponent      result
0000 92      :
0000 93      :   any      > 0          product (base * 2**i) where i is each
0000 94      :                                     non-zero bit position in exponent
0000 95      :
0000 96      :   > 0      = 0          1.0
0000 97      :   = 0      = 0          Undefined exponentiation
0000 98      :   < 0      = 0          1.0
0000 99      :
0000 100     :
0000 101     : CALLING SEQUENCE:
0000 102     :
0000 103     :   result.wg.v = OTSS$POWGLU (base.rg.v, exponent.rlu.v)
0000 104     :
0000 105     : FORMAL PARAMETERS:
0000 106     :
0000 107     :
00000004 0000 108     :   base = 4      ; G_floating base. Note that this is passed by
0000 109     :               ; immediate value in two argument list positions,
0000 110     :               ; which is a violation of the calling standard.
0000 111     :               ; This is allowed for language support procedures.
0000000C 0000 112     :
0000 113     :   exponent = 12 ; Unsigned longword integer exponent
0000 114     :
0000 115     :
0000 116     : IMPLICIT INPUTS:
0000 117     :
0000 118     :   NONE
0000 119     :
0000 120     : IMPLICIT OUTPUTS:
0000 121     :
0000 122     :   NONE
0000 123     :
0000 124     : ROUTINE VALUE:
0000 125     :
0000 126     :   The base raised to the exponent's power.
0000 127     :
0000 128     : SIDE EFFECTS:
0000 129     :
0000 130     :   May signal:
0000 131     :       MTH$_FLOOVEMAT - floating overflow in Math Library
0000 132     :       MTH$_FLOUNDMAT - floating underflow in Math Library, only
0000 133     :                           if the caller has FU enabled.
0000 134     :       MTH$_UNDEXP    - undefined exponentiation, if both base and
0000 135     :                           exponent are zero.
0000 136     :
0000 137     : --
0000 138

```



```

001C 0000 139 .ENTRY OTSSPOWGLU, ^M<R2,R3,R4> ; Entry point
0002 140
6D 6A'AF 9E 0002 141 MOVAB B^EXC HANDLER, (FP) ; Translate exceptions to MTH$ errors
0040 8F B8 0006 142 BISPSW #PSL$M_FU ; Enable floating underflow detection
50 08 50FD 000A 143 MOVG #1, R0 ; R0/R1 = initial result
52 04 AC 50FD 000E 144 MOVG base(AP), R2 ; R2/R3 = base
54 0C AC D0 0013 145 MOVL exponent(AP), R4 ; R4 = exponent
06 12 0017 146 BNEQ BEGIN ; Skip if not zero
52 53FD 0019 147 TSTG R2 ; Is base also zero?
23 13 001C 148 BEQL UNDEFINED ; If so, error. Otherwise, answer is
001E 149 ; 1.0.
04 001E 150 RET ; With result of 1.0
001F 151
001F 152 ;+
001F 153 ; Do the first iteration here so that we can clear the high-order exponent
001F 154 ; bit. Afterwards, that bit will be zero so we don't have to worry about
001F 155 ; it.
001F 156 ;+
001F 157
001F 158 BEGIN:
04 54 00 E5 001F 159 BBCC #0, R4, 10$ ; Skip initial multiply
54 54 50 52 50FD 0023 160 MOVG R2, R0 ; New partial result
54 54 FF 8F 9C 0027 161 10$: ROTL #-1, R4, R4 ; Get new low exponent bit and clear
002C 162 ; high bit
12 13 002C 163 BEQL DONE ; Is that all? If so, we're done.
002E 164
002E 165 ;+
002E 166 ; Each time we get here we know that there's at least one more exponent bit
002E 167 ; left, so square our current power of the base.
002E 168 ;+
002E 169
002E 170 LOOP:
52 52 44FD 002E 171 MULG2 R2, R2 ; Square base. This may overflow or
0032 172 ; underflow. In either case, the
0032 173 ; final result would also overflow
0032 174 ; or underflow. Our exception
0032 175 ; handler will catch these cases.
04 54 E9 0032 176 BLBC R4, NEXT_BIT ; See if this exponent bit is on.
0035 177 ; If so, multiply base by product.
50 52 44FD 0035 178 MULG2 R2, R0 ; This too may overflow or underflow,
0039 179 ; which will be caught by the handler.
0039 180 NEXT_BIT:
54 54 FF 8F 78 0039 181 ASHL #-1, R4, R4 ; Get next bit of exponent.
EE 12 003E 182 BNEQ LOOP ; Keep going if not all done.
0040 183
0040 184 DONE:
04 0040 185 RET ; Return with answer in R0/R1.
0041 186
0041 187 ;+
0041 188 ; Undefined exponentation error - 0**0
0041 189 ;+
0041 190
0041 191 UNDEFINED:
7E 00'8F 9A 0041 192 MOVZBL #MTH$K_UNDEXP, -(SP) ; Indicate undefined exponentiation.
50 01 0F 79 0045 193 ASHQ #15, #T, R0 ; R0/R1 = reserved floating operand
15 11 0049 194 BRB SIGNAL_ERROR
004B 195

```

```
004B 196 ;+
004B 197 ; EXC_HANDLER unwinds here if we got a floating overflow.
004B 198 ; -
004B 199
004B 200 OVERFLOW:
50 7E 01 00'8F 9A 004B 201 MOVZBL #MTH$K_FLOOVEMAT, -(SP) ; Floating overflow in Math Library
79 004F 202 ASHQ #15, #T, R0 ; R0/R1 = reserved floating operand
11 0053 203 BRB SIGNAL_ERROR
0055 204
0055 205 ;+
0055 206 ; EXC_HANDLER unwinds here if we got a floating underflow.
0055 207 ; If our caller enabled FU, then signal FLOUNDMAT, with a result of zero.
0055 208 ; Otherwise quietly return zero.
0055 209 ; -
0055 210
0055 211 UNDERFLOW:
E4 04 AD 50 7C 0055 212 CLRG R0 ; Initial result
06 E1 0057 213 BBC #PSL$V_FU, SF$W_SAVE_PSW(FP), DONE ; Just return if
005C 214 ; caller disabled underflow.
7E 00'8F 9A 005C 215 MOVZBL #MTH$K_FLOUNDMAT, -(SP) ; Floating underflow in Math Library
0060 216 ; BRB SIGNAL_ERROR
0060 217
0060 218 ;+
0060 219 ; Signal a MTH$ error whose code has already been pushed on the stack.
0060 220 ; R0/R1 has either a reserved operand or a zero, depending on the
0060 221 ; exception type.
0060 222 ; -
0060 223
0060 224 SIGNAL_ERROR:
00000000'GF 6D D4 0060 225 CLRL (FP) ; Cancel our condition handler
01 FB 0062 226 CALLS #1, G^MTH$$SIGNAL ; convert to 32-bit condition code
0069 227 ; and signal a MTH$ error. Omit
0069 228 ; second argument to show that this
0069 229 ; is a CALL entry point.
04 0069 230 RET ; Return to caller with result in
006A 231 ; R0/R1
006A 232
```



```
006A 234 ;+
006A 235 ; EXC_HANDLER - This condition handler gets control if we got an exception
006A 236 ; while performing the exponentiation. If it is a floating overflow,
006A 237 ; then continue execution at OVERFLOW, to cause MTH$_FLOOVEMAT to be
006A 238 ; signalled. Similarly, continue at UNDERFLOW if an underflow was seen.
006A 239 ; Otherwise resignal.
006A 240 ;--
006A 241
006A 242 EXC_HANDLER:
0000 006A 243 .WORD ^M<> ; Entry point
006C 244
50 08 AC D0 006C 245 MOVL CHF$_MCHARGLIST(AP), R0 ; Get mechanism arguments list
08 08 A0 D5 0070 246 TSTL CHF$_MCH_DEPTH(R0) ; At depth zero?
16 12 0073 247 BNEQ RESIGNAL ; if not, resignal
50 04 AC D0 0075 248 MOVL CHF$_SIGARGLIST(AP), R0 ; Get signal arguments list
51 04 A0 D0 0079 249 MOVL CHF$_SIG_NAME(R0), R1 ; Get signal name
04B4 8F 51 B1 007D 250 CMPW R1, #SS$_FLT_OVF_F ; Overflow fault?
0D 13 0082 251 BEQL GOTO_OVERFLOW ; If so, continue at OVERFLOW
04C4 8F 51 B1 0084 252 CMPW R1, #SS$_FLT_UND_F ; Underflow fault?
11 13 0089 253 BEQL GOTO_UNDERFLOW ; If so, continue at UNDERFLOW
008B 254 RESIGNAL:
50 0918 8F 3C 008B 255 MOVZWL #SS$_RESIGNAL, R0 ; Resignal exception
04 0090 256 RET ; Return to VMS exception dispatcher
0091 257
0091 258 GOTO_OVERFLOW:
FC A041 51 60 D0 0091 259 MOVL CHF$_SIG_ARGS(R0), R1 ; Get number of signal arguments
B4 AF DE 0094 260 MOVAL B^OVERFLOW, -4(R0)[R1] ; Move address of OVERFLOW routine
09 11 009A 261 BRB CONTINUE ; to PC in signal argument list
009C 262 CONTINUE ; Continue execution
009C 263
009C 264 GOTO_UNDERFLOW:
FC A041 51 60 D0 009C 265 MOVL CHF$_SIG_ARGS(R0), R1 ; Get number of signal arguments
B3 AF DE 009F 266 MOVAL B^UNDERFLOW, -4(R0)[R1] ; Move address of UNDERFLOW routine
00A5 267 ; to PC in signal argument list
00A5 268 ; BRB CONTINUE ; Continue execution
00A5 269
00A5 270 CONTINUE:
50 01 D0 00A5 271 MOVL #SS$_CONTINUE, R0 ; Continue at PC in signal list
04 00A8 272 RET ; Return to VMS exception dispatcher
00A9 273
00A9 274
00A9 275 .END ; End of module OTSS$POWGLU
```

OTSSPOWGLU
Symbol table

M 1
- G_floating ** unsigned integer power r 16-SEP-1984 02:00:02 VAX/VMS Macro V04-00 Page 7
6-SEP-1984 11:28:18 [MTHRTL.SRC]OTSPOWGLU.MAR;1 (4)

BASE	= 00000004		
BEGIN	0000001F	R	02
CHFSL_MCHARGLST	= 00000008		
CHFSL_MCH_DEPTH	= 00000008		
CHFSL_SIGARGLST	= 00000004		
CHFSL_SIG_ARGS	= 00000000		
CHFSL_SIG_NAME	= 00000004		
CONTINUE	000000A5	R	02
DONE	00000040	R	02
EXC_HANDLER	0000006A	R	02
EXPONENT	= 0000000C		
GOTO_OVERFLOW	00000091	R	02
GOTO_UNDERFLOW	0000009C	R	02
LOOP	0000002E	R	02
MTHSSIGNAL	*****	X	00
MTHSK_FLOOVEMAT	*****	X	00
MTHSK_FLOUNDMAT	*****	X	00
MTHSK_UNDEXP	*****	X	00
NEXT_BIT	00000039	R	02
OTSSPOWGLU	00000000	RG	02
OVERFLOW	0000004B	R	02
PSLSM_FU	= 00000040		
PSLSV_FU	= 00000006		
RESIGNAL	0000008B	R	02
SFSW_SAVE_PSW	= 00000004		
SIGNAL_ERROR	00000060	R	02
SSS_CONTINUE	= 00000001		
SSS_FLT0VF_F	= 000004B4		
SSS_FLTUND_F	= 000004C4		
SSS_RESIGNAL	= 00000918		
UNDEFINED	00000041	R	02
UNDERFLOW	00000055	R	02

+-----+
! Psect synopsis !
+-----+

PSECT name	Allocation	PSECT No.	Attributes														
. ABS .	00000000 (0.)	00 (0.)	NOPI	USR	CON	ABS	LCL	NOSHR	NOEXE	NORD	NOWRT	NOVEC	BYTE				
\$ABSS	00000000 (0.)	01 (1.)	NOPI	USR	CON	ABS	LCL	NOSHR	EXE	RD	WRT	NOVEC	BYTE				
_OTSSCODE	000000A9 (169.)	02 (2.)	PIC	USR	CON	REL	LCL	SHR	EXE	RD	NOWRT	NOVEC	LONG				

+-----+
! Performance indicators !
+-----+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	31	00:00:00.05	00:00:00.99
Command processing	132	00:00:00.47	00:00:02.68
Pass 1	208	00:00:04.84	00:00:13.57
Symbol table sort	0	00:00:00.77	00:00:01.24
Pass 2	62	00:00:01.07	00:00:03.44
Symbol table output	5	00:00:00.05	00:00:00.08
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00

Assembler run totals 442 00:00:07.27 00:00:22.11

The working set limit was 1200 pages.

26459 bytes (52 pages) of virtual memory were used to buffer the intermediate code.

There were 30 pages of symbol table space allocated to hold 508 non-local and 1 local symbols.

275 source lines were read in Pass 1, producing 13 object records in Pass 2.

11 pages of virtual memory were used to define 10 macros.

+-----+
! Macro library statistics !
+-----+

Macro library name

Macros defined

_S255SDUA28:[SYSLIB]STARLET.MLB;2

7

562 GETS were required to define 7 macros.

There were no errors, warnings or information messages.

MACRO/ENABLE=SUPPRESSION/DISABLE=(GLOBAL,TRACEBACK)/LIS=LIS\$:OTSPOWGLU/OBJ=OBJ\$:OTSPOWGLU MSRC\$:OTSPOWGLU/UPDATE=(ENH\$:OTSPOWGLU)

0265 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

